



Create REST Web Services from RPG Programs (and SQL!)

Jon Paris

Jon.Paris @ Partner400.com
www.Partner400.com

*Paris
Partner400*

Notes



About Jon Paris (that's me):

My career in IT spans 50+ years including a 12 year period with IBM's Toronto Laboratory. One of my proudest achievements during that period was being a founding member of the small team that designed and built RPG IV.

I am the co-founder of Partner400, a firm specializing in customized education and mentoring services for IBM i (AS/400, System i, iSeries, etc.) developers. I am also a founding member of the System i Developer education consortium which, among other things organized the twice yearly RPG & DB2 Summit.

Together with my partner Susan Gantner, I devote my time to educating developers on techniques and technologies to extend and modernize their applications and development environments. We also write frequently for IT Jungle's RPG Guru column (www.itjungle.com). We have also written many times for IBM Systems Magazine and other publications. You can find our "collected works" at our Authority site <https://authority.com/jonparisandsusangantner>

Feel free to contact me any time: [Jon.Paris @ partner400.com](mailto:Jon.Paris@partner400.com)

Creating REST Web Services - Basics

REST = Representational State Transfer

- Doesn't really explain a lot though does it !

Basically a URL that identifies a processing resource

- Think of it as being a request for a "web page" designed to be filled in or consumed by a program

The HTTP request type differentiates the required actions

- GET retrieves information
 - ✦ This is the one we will use - but IWS supports others too
- PUT is used to create a new item
- POST will completely update an item (or create it if it does not exist)
- There's also PATCH (Partial Update) and DELETE

Input and output data can be in any format

- For REST services input data is frequently part of the URL &/or the query string
 - ✦ If in the body it is most commonly coded as JSON or XML

IBM supply us with the Integrated Web Services Server (IWS)

- This will be the tool I will be looking at today
- It provides an easy-to-use option for quickly deploying web services
 - ✦ Using regular programs or SQL statements for the processing

IBM's IWS - ibm.com/systems/i/software/iws/

*Paris
Gartner400*

Available for several years now

- I first wrote about it back in 2012

Originally we recommended clients use it only for Proof of Concept

- i.e. to demonstrate that IBM i could actually play in the web service arena

Over time it has improved significantly

- And can now be a good candidate for production workloads

Turns a simple *PGM, *SRVPGM or SQL statement into a Web Service

- The IBM supplied wizard does all the work

IWS can create SOAP or REST services

- And is frequently enhanced by IBM

For details go to:

- ibm.com/support/pages/integrated-web-services-ibm-i-web-services-made-easy
 - ✦ Or simply Google IWS Home Page site:IBM.com
- You'll find the documentation links at the foot of the page

But before you can start using it ...

IWS Pre-Requisites and Documentation

Partner400

These products are required - But you'll almost certainly have them installed already

- Qshell - base option 30 of operating system
- PASE - base option 33 of operating system
- Host Servers - base option 12 of operating system
- Digital Certificate Manager - base option 34 of operating system
- IBM HTTP Server for IBM i

Optionally:

- IBM Java SE 7 32 bit must be loaded for Qshell script support
 - ✦ IBM recommend loading Java SE 8 64 bit
 - That ensures that any servers that are created use the latest supported runtime.

Make sure all current group PTFs are applied

- **Particularly the HTTP Server PTFs**
 - ✦ New features are usually added to IWS via PTFs

Documentation

- Integrated Web Services Server - Administration and Programming Guide
 - ✦ 2018 version on web - not completely up-to-date
 - Links are on the IWS main page as noted on previous page
 - ✦ For generating IWS services using SQL see:
 - developer.ibm.com/tutorials/creating-rest-apis-based-on-sql-statements/

Getting Started ...

Begin by connecting to the Admin server:

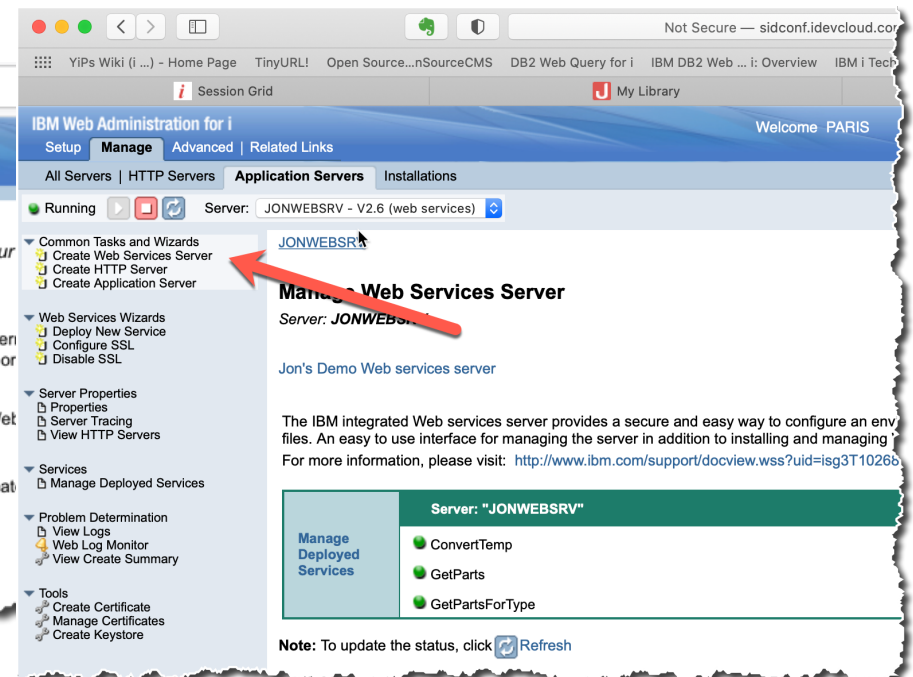
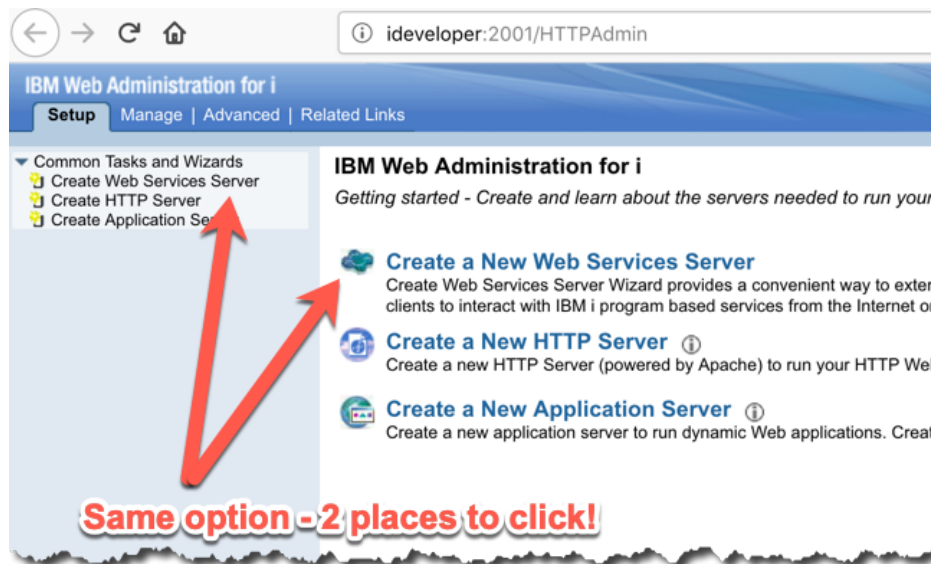
- <http://YourServerName:2001/HTTPAdmin>

You will be asked to sign in and one of the screens below will eventually appear:

- If you have previously defined web servers you will see the screen on the right
 - ✦ Otherwise you should see the one on the left

Click on "Create ... Web Services Server"

- Note that there is no difference between the links shown
- They will all take you to the same place



Name Your Server

Every server needs a name

- This can be any name of your choosing
 - ✦ But make it meaningful

Enter a name and then press "Next"

- You can also enter a description if you wish
 - ✦ Probably a good idea for a production server

IBM Web Administration for i Welcome PARIS

Setup **Manage** Advanced | Related Links

All Servers | HTTP Servers **Application Servers** Installations

Running ▶ ■ ↻ Server: JONWEBSRV - V2.6 (web services) ▾

Create Web Services Server
Specify Web services server name - Step 1 of 4

Welcome to the Create Web Services Server wizard. A Web services server provides a secure and es...
programs and SQL statements. This wizard creates everything needed to run Web services.
For more information, please visit: <http://www.ibm.com/support/docview.wss?uid=isg3T1026868>

Specify a unique name for this server ?

Server name: LandL

Server description: Web Services Server for Lunch & Learn

Create HTTP server

De-select if no regular web pages are to be served and Basic HTTP authentication is not needed

Identify the Ports to use

Each server uses two ports

- One for managing the server (10023 in this example)
- And a second which is the actual service point (10022)

IWS automatically selects ports for you

- But you can change them if needed
 - ✦ e.g. To match open ports on your firewall



Server Operating Environment

Paris Gartner400

Releases 7.3+ allow you to control the server subsystem etc.

- If you take the defaults it will run in QHTTPSVR
 - ✦ Earlier releases don't show this screen and will default to QHTTPSVR

IBM Web Administration for i Welcome PARIS

Setup **Manage** Advanced | Related Links

All Servers | HTTP Servers **Application Servers** Installations

Running ▶ ■ ↺ Server: JONWEBSRV - V2.6 (web services) ▾

Create Web Services Server
Specify subsystem for server - Step 3 of 5

Specify the operating environment for the server's jobs by specifying work management attributes the controls who server jobs running in subsystem QHTTPSVR. ?

Path to job description: /QSYS.LIB/QHTTPSVR.LIB/QZHBHTTP.JOBD or... ▾ Browse

Path to job queue: *JOBQ or... ▾ Browse

Routing data: *JOBQ or... ▾

I will be simply using the defaults

Select the User Profile

For proof of concept purposes the default user works fine

- For production it is best to create a new user that only has the needed authorities
- The manual describes the minimum authorities needed
 - ✦ Make sure the user is also authorized to any libraries needed by your service

ideveloper:2001/HTTPAdmin

IBM Web Administration for i

Setup | Manage | Advanced | Related Links

Common Tasks and Wizards

- Create Web Services Server
- Create HTTP Server
- Create Application Server

Create Web Services Server

Specify User ID for Server - Step 2 of 3

The server requires an IBM i user ID to run the server's jobs. It is recommended that a user ID be specified to run the server's jobs since this user ID is given authority to all of the server's jobs and directories.

Specify user ID for this server: ?

Use default user ID

Note: The default server user ID is QWSERVICE.

Specify an **existing** user ID

Create a **new** user ID

Back Next Cancel

First Steps - Completing the Task

This page summarizes what the wizard is building for you

- Note the Server URL
- And the context root

Press "Finish"

- This will actually build the server
- And start it running

The screenshot shows the IBM Web Administration console interface. The browser address bar indicates the URL is `sidconf.idevcloud.com:2001/HTTPAdmin`. The page title is "IBM Web Administration for i". The navigation tabs include "Setup", "Manage", "Advanced", and "Related Links". The main content area is titled "Create Web Services Server" and shows the "Summary - Step 4 of 4". The "Web Services Server Information" section lists the following details:

- Server name: MSPTEST
- Server description: Web services server created by the Create Web Services Server wizard.
- Port: 10025
- Command port: 10026
- Server root: /www/MSPTEST
- Server URL: `http://sidc.idevcloud.com:10035` (indicated by a red arrow)
- User ID for server: QWSERVICE
- Context root: /web (indicated by a red arrow)

The "HTTP Server Information" section lists the following details:

- HTTP server name: MSPTEST
- HTTP server description: Web services server created by the Create Web Services Server wizard.
- Port: 10035
- Document root: /www/MSPTEST/htdocs
- Server root: /www/MSPTEST
- Server association: MSPTEST

At the bottom of the wizard, there are three buttons: "Back", "Finish", and "Cancel".

First Steps - Progress Display

This page appears during the build process

- You normally have to press the Refresh button to see the server page
 - ✦ I have never been able to work out when it does it automatically and when you have to manually request a refresh

The screenshot shows the IBM Web Administration for i console. The browser address bar indicates the URL is sidconf.idevcloud.com:2001/HTTPAdmin. The console has tabs for Setup, Manage, Advanced, and Related Links. Under the Manage tab, there are sub-tabs for All Servers, HTTP Servers, Application Servers, and Installations. The Application Servers tab is active, and a dropdown menu shows 'Server: MSPTEST - V2.6 (web services)'. In the left sidebar, under 'Common Tasks and Wizards', there are three options: 'Create Web Services Server', 'Create HTTP Server', and 'Create Application Server'. A red arrow points to the 'Create Web Services Server' option. A callout box with a red border and white text says 'Press this to refresh the page'. The main content area shows the 'Manage Web Services Server' page for 'MSPTEST'. It states 'Server: MSPTEST' and 'Web services server created by the Create Web Services Server wizard.' Below this, there is a note: 'The IBM integrated Web services server provides a secure and easy way to configure an environment for hosting Web services. An easy to use interface for managing the server in addition to installing and managing Web services is provided. For more information, please visit: <http://www.ibm.com/support/docview.wss?uid=isg3T1026868>'. At the bottom, there is a status box: 'Server "MSPTEST" is in the process of being created. To update the status, click the Refresh icon above.' Below this, a note says: 'Note: To update the status, click Refresh'.

First Steps - The Finished Server

Paris
Gartner400

A sample SOAP service has been deployed for you

- If you wish you can use it to test the basic server operation
 - ✦ I normally just delete it

To deploy a new service we need to "Manage ..." the services

- So click the marked big green box

IBM Web Administration for i

Setup **Manage** Advanced | Related Links

All Servers | HTTP Servers **Application Servers** Installations

Running Server: MSPTTEST - V2.6 (web services)

Server controls

Manage Web Services Server

Server: MSPTTEST

Web services server created by the Create Web Services Server wizard.

The IBM integrated Web services server provides a secure and easy way to configure... An easy to use interface for managing the server in addition to installing and managing... For more information, please visit: <http://www.ibm.com/support/docview.wss?uid=isg>

Server: "MSPTTEST"	
Manage Deployed Services	<input type="checkbox"/> ConvertTemp

Note: To update the status, click Refresh

List of deployed services and their status

Click here to manage the services

Service Management

Use this to:

- Create new services (Deploy)
 - ✦ And to Modify (Redeploy), Stop, Start, and Uninstall them
- You can also review a service's properties

We'll just create (Deploy) a new one

The screenshot shows the IBM Web Administration for i console. The top navigation bar includes 'Setup', 'Manage', 'Advanced', and 'Related Links'. The 'Manage' tab is active. Below the navigation bar, there are tabs for 'All Servers', 'HTTP Servers', 'Application Servers', and 'Installations'. The 'Application Servers' tab is selected. The main content area shows the 'Manage Deployed Services' page for the 'LandL - V2.6 (web services)' server. The page title is 'LandL > Manage Deployed Services'. The data is current as of Mar 8, 2021 12:54:36 PM. There is one deployed service listed in a table:

Service name	Status	Type	Startup type	Service definition
ConvertTemp	Running	SOAP	Automatic	View WSDL

Below the table, there are several buttons: 'Deploy', 'Stop', 'Properties', 'Uninstall', 'Redeploy', and 'Refresh'. The 'Deploy' button is highlighted with a red circle.

Deploying SQL as a Web Service

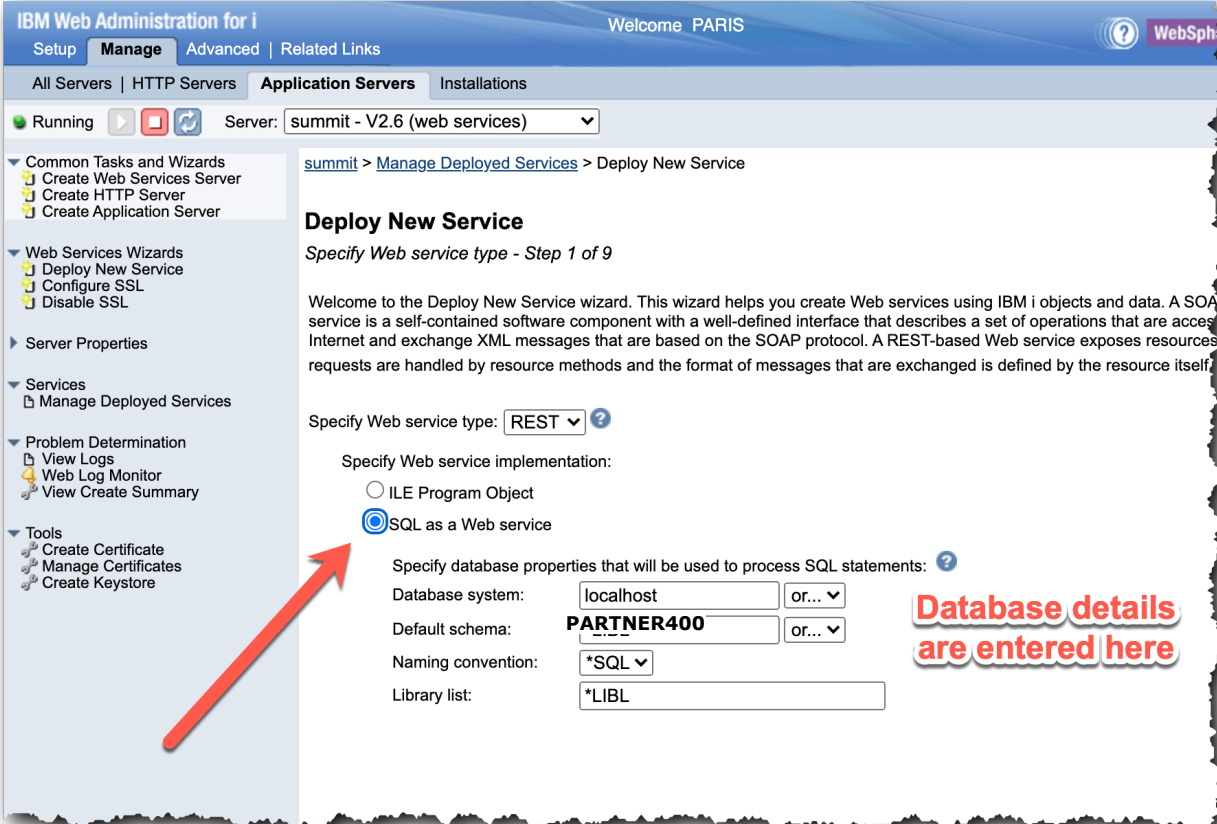
Paris
Partner400

Your SQL skills determine how complex a response can be built

- You will see how the SQL is coded in a moment

You can specify a library (schema) to use as the default

- That way you don't always have to fully qualify the table names



IBM Web Administration for i

Welcome PARIS

Setup Manage Advanced | Related Links

All Servers | HTTP Servers Application Servers Installations

Running Server: summit - V2.6 (web services)

summit > Manage Deployed Services > Deploy New Service

Deploy New Service

Specify Web service type - Step 1 of 9

Welcome to the Deploy New Service wizard. This wizard helps you create Web services using IBM i objects and data. A SOA service is a self-contained software component with a well-defined interface that describes a set of operations that are accessed over the Internet and exchange XML messages that are based on the SOAP protocol. A REST-based Web service exposes resources and requests are handled by resource methods and the format of messages that are exchanged is defined by the resource itself.

Specify Web service type: REST

Specify Web service implementation:

- ILE Program Object
- SQL as a Web service

Specify database properties that will be used to process SQL statements:

Database system: localhost or...

Default schema: PARTNER400 or...

Naming convention: *SQL

Library list: *LIBL

Database details are entered here

Deploying an SQL Service (Cont)

Here we get to supply the name for the service

- GetPartsForCat

And specify how the requested parameter will be passed

- In this case in the URL immediately following the last /
- You will see how to associate the name {cat} with the data in a moment

IBM Web Administration for i Welcome PARIS

Setup **Manage** Advanced | Related Links

All Servers | HTTP Servers **Application Servers** Installations

Running ▶ ■ ↺ Server: JONWEBSRV - V2.6 (web services) ▾

LandL > Manage Deployed Services > Deploy New Service

Deploy New Service

Specify Name for Service - Step 2 of 9

The Web service to be externalized is a resource. The URI path template identifies matching patterns for incoming parameters that can contain regular expressions to further restrict what is allowed. ?

Resource name:

Service description:

URI path template: e.g. /temperature, /temperature/{temp:d+}

We will use {cat} in a moment

Deploying an SQL Service (Cont)

Time to enter the SQL statement(s)

Enter the SQL using the standard "?" as a parameter marker

- The wizard makes up a name for the parameter
 - ✦ We will change it to a meaningful name next

Use the **Continue** button when all SQL has been entered

- or use the **Add** button if you need additional SQL statements

IBM Web Administration for i
Setup | **Manage** | Advanced | Related Links
Welcome PARIS

All Servers | HTTP Servers | **Application Servers** | Installations

Running | Server: JONWEBSRV - V2.6 (web services)

Common Tasks and Wizards
Create Web Services Server
Create HTTP Server
Create Application Server

Web Services Wizards
Deploy New Service
Configure SSL
Disable SSL

Server Properties
Properties
Server Tracing
View HTTP Servers

Services
Manage Deployed Services

Problem Determination
View Logs
Web Log Monitor
View Create Summary

Tools
Create Certificate
Manage Certificates
Create Keystore

LandL > Manage Deployed Services > Deploy New Service

Deploy New Service

Specify SQL Statements - Step 4 of 8

Specify SQL statements:

Procedure name	SQL Statement	Parameter name	Us
<input checked="" type="checkbox"/> PartsForCat	select * from product where catcod = ?		

Add Remove Remove All **Continue** Insert from example

Deploying an SQL Service (Cont)

Once the Continue button has been pressed this appears

- Notice the ugly name **PARM00001**

To change it select the procedure

- The second screen (which looks much like the original) will appear
 - ✦ But this time you can change the parameter name

Deploy New Service

Specify SQL Statements - Step 4 of 8

Specify SQL statements: ?

	Procedure name	SQL statement/Parameter name	Usage	Da
<input type="checkbox"/>	GetPartsForCat	select * from product where catcod = ?		
		PARM00001	input	CH

Add Remove All

Deploy New Service

Specify SQL Statements - Step 4 of 8

Specify SQL statements: ?

	Procedure name	SQL statement/Parameter name
<input checked="" type="checkbox"/>	GetPartsForCat	select * from product where catcod = ?

Category

Add Remove Remove All Continue Insert from example

Deploying an SQL Service (Cont)

You can specify single row

- Or multi-row result

Trimming options on results

What constitutes an error

- Specify a custom error message

What HTTP status to return

Etc. etc.


JPRESTSVR > [Manage Deployed Services](#) > Deploy New Service


Deploy New Service


Specify SQL Information - Step 5 of 8


Customize how each procedure will process SQL statements. For query statements, this includes:

Procedure name: ProductSelect
SQL Statement: select * from product where catcod = ?


SQL result type: Multi-row result set 


Trim mode for output fields: Trailing 

SQL state information in response: On errors 

Treat warnings as SQL Errors: Yes 

User-defined error message:

HTTP status code on SQL success: 200 or... 

HTTP status code on SQL failure: 500 or... 

Deploying an SQL Service (Cont)

To assign the input parameter ...

- For the parameter Category select *PATH_PARAM as the source
- The identifier "cat" will appear

That's all we have to do

- The defaults for everything else are fine at this stage

Deploy New Service
Specify Resource Method Information - Step 6 of 8

Procedures are mapped to resource methods. Each resource method needs to be defined to handle client requests by mapping an HTTP request method.

Procedure name: GetPartsForCat
URI path template for resource: /(cat)
HTTP request method: GET
URI path template for method: *NONE or...
HTTP header information: *NONE
Allowed input media types: *ALL or...
Returned output media types: *JSON
Identifier for input wrapper element: GetPartsForCatInput or...
Identifier for output wrapper element: GetPartsForCatResult or...
Whether to wrap input parameters:
 Wrap input parameters
 Do not wrap input parameters

Input parameter mappings:

Parameter name	Data type	Input source	Identifier	Default Value
Category	CHAR	*PATH_PARAM or...	cat or...	*NONE or...

Back Next Cancel

JSON is selected by default, but we could specify XML, HTML, or any combination

Deploying an SQL Service (Cont)

Last but not least specify the User Id to be used

- In most cases the default will be fine
- To use "authenticated" requires that you specified an HTTP server during the App Server configuration earlier

The screenshot shows the IBM Web Administration for i console. The top navigation bar includes 'Setup', 'Manage', 'Advanced', and 'Related Links'. Below this, there are tabs for 'All Servers', 'HTTP Servers', 'Application Servers', and 'Installations'. The 'Application Servers' tab is active, and the server 'JONWEBSRV - V2.6 (web services)' is selected. The left sidebar contains a tree view with categories like 'Common Tasks and Wizards', 'Web Services Wizards', 'Server Properties', 'Services', 'Problem Determination', and 'Tools'. The main content area shows the 'Deploy New Service' wizard, specifically 'Specify User ID for this Service - Step 7 of 8'. The text indicates that the service requires an IBM i user ID. Three radio button options are provided: 'Use server's user ID' (selected), 'Specify an existing user ID', and 'Use authenticated user ID'.

Deploying an SQL Service (Cont)

This are the three tabs of the final confirmation screen

- You can use it to check that your details are correct

But in a simple case like this I will just press **Finish**

- That will install and launch the new service

LandL > Manage Deployed Services > Deploy New Service

Deploy New Service

Summary - Step 8 of 8

When you click **Finish** the web service is deployed.

Service | JDBC Properties | Methods

Resource name: GetPartsForCat
Resource description: Get All Parts for Category
Service install path : /www/landl/webservices/services/
URI path template: /{cat}
User ID for service: *SERVER (QWSERVICE)

LandL > Manage Deployed Services > Deploy New Service

Deploy New Service

Summary - Step 8 of 8

When you click **Finish** the web service is deployed.

Service | JDBC Properties | Methods

Database system: localhost
Default schema: partner400
Naming convention: *SQL
Library List: *LIBL

LandL > Manage Deployed Services > Deploy New Service

Deploy New Service

Summary - Step 8 of 8

When you click **Finish** the web service is deployed.

Service | JDBC Properties | Methods

Procedure name: GetPartsForCat
SQL Statement: select * from product where catcod = ?
SQL result type: Multi-row result set
Trim mode for output fields: Trailing
SQL state information in response: On errors
Treat warnings as SQL Errors: Yes
User-defined error message:
HTTP status code on SQL success: 200
HTTP status code on SQL failure: 500
HTTP request method: GET
URI path template for method: *NONE
HTTP header information: *NONE
Allowed input media types: *ALL
Returned output media types: *JSON
Identifier for input wrapper element: GetPartsForCatInput
Identifier for output wrapper element: GetPartsForCatResult

Input parameter mappings:

Parameter name	Data type	Input source	Identifier	Default Value
Category	CHAR	*PATH_PARAM	cat	*NONE

Deploying an SQL Service (Cont)

Paris
Gartner400

Invoking the new REST service from the browser using
sidconf.idevcloud.com:10006/web/services/GetPartsForCat/02

Produces this result

- Note the name given to the resulting JSON array



```
{
  "GetPartsForCat_GetPartsForCat_R" : [
    { "PRODCD" : "000008", "PRODDS" : "Multi Purpose Spanner", "CATCOD" : "02", "STOH" : 50, "LNCST" : 0.75,
      "SELLPR" : 2.50, "DTLCHG" : 105, "DTLORD" : 990105},
    { "PRODCD" : "100001", "PRODDS" : "Another new one", "CATCOD" : "02", "STOH" : 5, "LNCST" : 100.00, "SELLPR" :
      200.00, "DTLCHG" : 0, "DTLORD" : 0},
    { "PRODCD" : "100002", "PRODDS" : "Yet another new one", "CATCOD" : "02", "STOH" : 5, "LNCST" : 100.00,
      "SELLPR" : 200.00, "DTLCHG" : 0, "DTLORD" : 0},
    { "PRODCD" : "200001", "PRODDS" : "Two Handed Spanner", "CATCOD" : "02", "STOH" : 200, "LNCST" : 95.00,
      "SELLPR" : 100.50, "DTLCHG" : 215, "DTLORD" : 104},
    { "PRODCD" : "200004", "PRODDS" : "Left Hand", "CATCOD" : "02", "STOH" : 100, "LNCST" : 11195.00, "SELLPR" :
      100.50, "DTLCHG" : 215, "DTLORD" : 104}
  ]
}
```

Writing Your Own Web Service Code

My sample program is a very simple one

- It accepts a product category as an input parameter
- And returns the details of all the products in that category
 - ♦ Plus a message containing a count of the number of products

There are 2 versions

- The first returns a count of the number of entries plus the data
 - ♦ As you will see it may return blank entries depending on how the service is set up
- The second uses a count to limit the number of entries
 - ♦ It wo'nt return the actual count value

In both cases the parameter list is used to drive the interface

- The data is returned in a DS
 - ♦ This is the normal approach for this type of service

The next chart details the parameter list

- And addresses other essential settings in the program source

The RPG Program - Version 1

The actual logic I used is not terribly important

- If you care ... it is a simple SQL cursor loop
- The program parameters are the important thing

Plus you need these CTL-OPT (H-spec) options you probably don't normally use

- PGMINFO(*PCML : *MODULE)
 - ✦ This tells the compiler to store parameter details in the module
 - ✦ IWS will extract this information to build the interface
- DECEDIT('0.')
 - ✦ Prevents RPG from suppressing the leading zero before a decimal points
 - ✦ We need this because '.5' is not a valid JSON number - it must be '0.5'

```
ctl-opt PGMINFO( *PCML : *MODULE ) DECEDIT('0.');
```

```
dcl-pi RestSrv4 ExtPgm;  
    requestedCat      Like(catcod);  
    result            char(20);  
    productList_count int(5);  
    productList      LikeDS(product) Dim(20);  
end-pi;
```

```
Dcl-ds  product ExtName('PRODUCT')  End-ds;
```

Writing Your Own - Version 1 (Contd)

First parameter is the requested category (input)

- Extracted from the URL by the IWS routines

The second is for an output text message from the program

- "No data found" or similar

The third is the count of the number of products returned

- Later versions move this into the list DS
 - ✦ You'll see why in a minute

Fourth one is the list of products

```
ctl-opt PGMINFO( *PCML : *MODULE ) DECEDIT('0.');
```

```
dcl-pi RestSrv4 ExtPgm;  
  requestedCat      Like(catcod);  
  result            char(20);  
  productList_count int(5);  
  productList       LikeDS(product) Dim(20);  
end-pi;
```

```
Dcl-ds  product ExtName('PRODUCT')  End-ds;
```

Note:
_count field is a
separate parameter

Service Management

OK time to create a REST service from an RPG program

- We start the process by pressing "Deploy"

IBM Web Administration for i
Setup **Manage** Advanced | Related Links

All Servers | HTTP Servers **Application Servers** | Installations

Running Server: JPRESTSVR - V2.6 (web services)

Common Tasks and Wizards
Create Web Services Server
Create HTTP Server
Create Application Server

Web Services Wizards
Deploy New Service
Configure SSL
Disable SSL

Server Properties

Services
Manage Deployed Services

Problem Determination
View Logs
Web Log Monitor
View Create Summary

Tools
Create Certificate
Manage Certificates
Create Keystore

JPRESTSVR > Manage Deployed Services

Manage Deployed Services

Data current as of Oct 16, 2019 11:20:21 AM.

Deployed services: ?

	Service name	Status	Type	Startup type	Service definition
<input type="radio"/>	ConvertTemp	Running	SOAP	Automatic	View WSDL
<input checked="" type="radio"/>	NEWSQL	Running	REST	Automatic	View Swagger
<input type="radio"/>	NewRESTSRV5	Running	REST	Automatic	View Swagger
<input type="radio"/>	RESTSRV4	Running	REST	Automatic	View Swagger
<input type="radio"/>	RESTSRV5	Running	REST	Automatic	View Swagger
<input type="radio"/>	SQL	Running	REST	Automatic	View Swagger

Deploy Stop Properties Uninstall Redeploy Refresh

Add a new service

Stop the service Required if you want to modify it

Uninstall service

Really means "Modify"

Deploying an RPG Service

On the first page you select the type of service

- REST (or if desired SOAP)

This time we specify implementation as being via a *PGM/*SRVPGM

- We also identify the path to the executable code

Note: I will not go through all the pages that work the same as in the SQL example

- I will just be highlighting the differences.

IBM Web Administration for i Welcome PARIS

Setup **Manage** Advanced | Related Links

All Servers | HTTP Servers **Application Servers** Installations

Running ▶ ⏏ ↺ Server: summit - V2.6 (web services) ▼

summit > Manage Deployed Services > Deploy New Service

Deploy New Service

Specify Web service type - Step 1 of 9

Welcome to the Deploy New Service wizard. This wizard helps you create Web services using IBM i objects and data. A SOAP-based Web service is a operations that are accessible over the Internet and exchange XML messages that are based on the SOAP protocol. A REST-based Web service exposes messages that are exchanged is defined in the resource itself. ?

Specify Web service type: **REST** ?

Specify Web service implementation:

ILE Program Object

Specify path to ILE program or service program: ?

Path of program object: e.g. /QSYS.LIB/MYLIB.LIB/MY.SRVPGM

Note: Specify a *PGM or *SRVPGM object.

SQL as a Web service

Type of service

Implementing Program/Service Program

Deploying an RPG Service (Contd.)

These were my initial parameter settings

- Note "Detect Field Lengths"

But using this caused the entire array to be output

- Seems backwards doesn't it

This caused of all the empty product entries

We will see in a moment what happens when that is deselected ...

Application Servers Installations

Source: JPRESTSVR - V2.6 (web services)

JPRESTSVR > Manage Deployed Services > Redeploy Service

Redeploy Service

Select Export Procedures to Externalize as a Web Service - Step 2 of 7

Exported procedures are entry points to a program object and are mapped to Web service operations. A service program contains one or more procedures. A program contains only one procedure.

The table below lists all the exported procedures found in the program object that can be externalized. The parameter attribute affects what data is sent by clients and what is returned by the Web service.

Detect length fields

Use parameter name as element name for data structures

Export procedures: ?

Select	Procedure name/Parameter name	Usage	Data type
<input checked="" type="checkbox"/>	RESTSRV4		
	REQUESTEDCAT	input	char
	RESULT	output	char
	PRODUCTLIST_COUNT	output	int
	PRODUCTLIST	output	struct

Select All Deselect All Expand All Collapse All

Deploying an RPG Service (Contd.)

This is basically what the returned data looks like

- We'll see this live in a moment

The important thing is that my code did not build this JSON

- The code generated by the IWS wizard did

```
{
  "RESULT": "Found 1 rows for 01",
  "PRODUCTLIST_COUNT": 1,
  "PRODUCTLIST": [
    {
      "PRODCD": "0000020",
      "PRODDS": "A duplicate test",
      "CATCOD": "01",
      "STOH": 456,
      "LNDCST": 20.00,
      "SELLPR": 30.00,
      "DTLCHG": 107,
      "DTLORD": 990105
    },
    {
      "PRODCD": "",
      "PRODDS": "",
      ...
    }
  ]
}
```

Data names are derived from the parameters

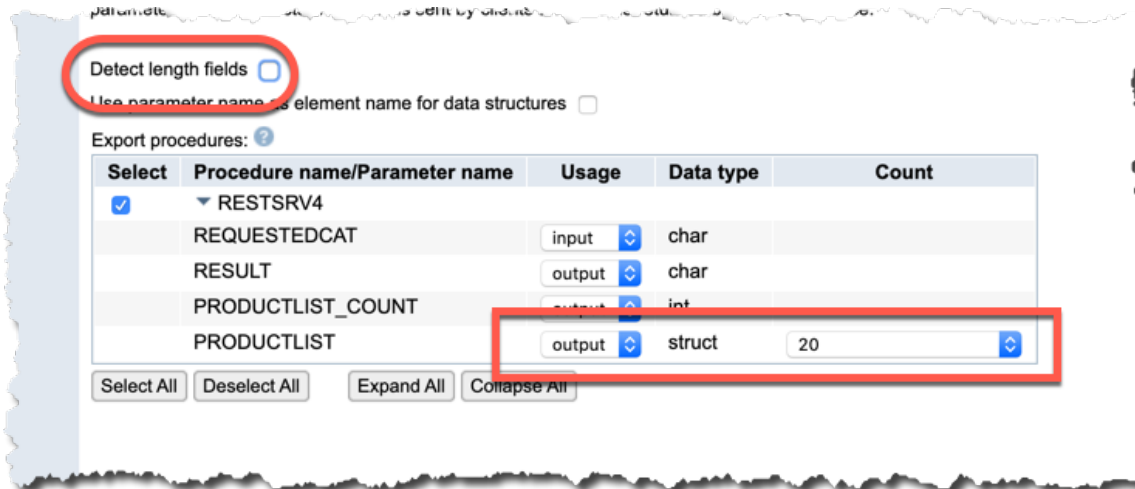
The _COUNT field is included in the output

The data types are also derived from the parameters. Note that the numeric fields are correctly represented

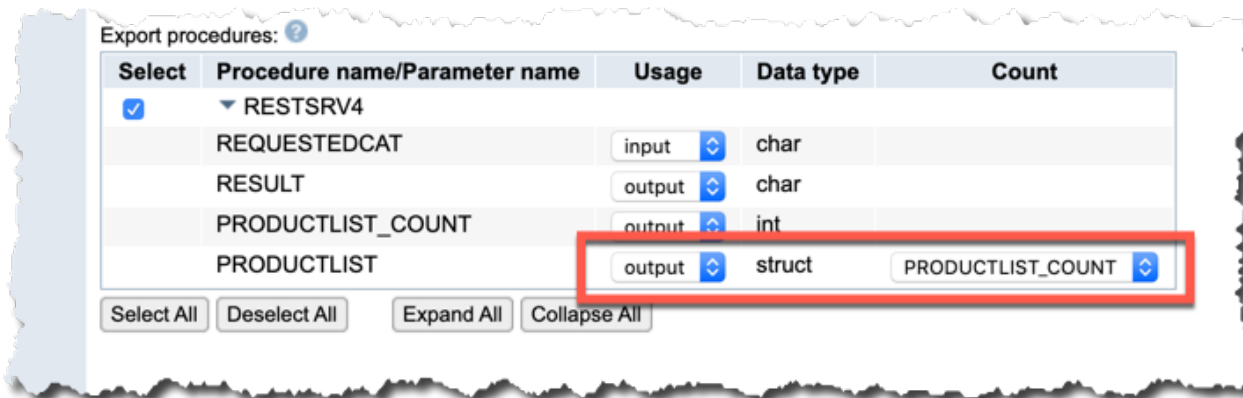
"Empty" entries. This is not normally desirable or acceptable

Deploying an RPG Service (Contd.)

If I deselect the "Detect" box a length selection option appears



One of the values that can be selected is PRODUCTLIST_COUNT



Let's see the result of making that change

Deploying an RPG Service (Contd.)

Notice the difference in the returned data

- No more empty entries
- Just the number specified by the count

In a moment we'll look at an alternative approach

- For cases where the count is not required as a value

```
{
  "RESULT": "Found 1 rows for 01",
  "PRODUCTLIST_COUNT": 1,
  "PRODUCTLIST": [
    {
      "PRODCD": "0000020",
      "PRODDS": "A duplicate test",
      "CATCOD": "01",
      "STOH": 456,
      "LNDCST": 20.00,
      "SELLPR": 30.00,
      "DTLCHG": 107,
      "DTLORD": 990105
    }
  ]
}
```

Array terminates after
the last item

Deploying an RPG Service - Version 2

Paris
Partner400

These are the parameter definitions for the second version

- Primary difference is in the positioning of the count
 - ✦ It is now WITHIN the result DS
- It must have the same base name as the array
 - ✦ The suffix _LENGTH tells the wizard to use it to control the returned length

Let's see how that affects the wizard and the subsequent results

```
ctl-opt PGMINFO(*PCML : *MODULE) DECEDIT('0.')

dcl-pi RestSrv5 ExtPgm;
  requestedCat      Like(catcod);
  result            char(20);
  productList      LikeDS(productList_T);
end-pi;

Dcl-ds  product ExtName('PRODUCT')  End-ds;

Dcl-ds  productList_T  Qualified Template;
  products_LENGTH      int(5);
  products              LikeDS(product)  Dim(20);
End-Ds;
```

Deploying an RPG Service - Version 2 (Contd.)

Paris Gartner400

Notice that this time we allow the wizard to detect the length

It will identify the field **products_LENGTH**

- And use it to control the number of **products** to output

IBM Web Administration for i | Welcome PARIS | WebSphere | IBM

Setup | **Manage** | Advanced | Related Links

All Servers | HTTP Servers | **Application Servers** | Installations

Running | Server: summit - V2.6 (web services)

summit > Manage Deployed Services > Redeploy Service

Redeploy Service

Select Export Procedures to Externalize as a Web Service - Step 2 of 7

Exported procedures are entry points to a program object and are mapped to Web service operations. A procedure is a set of self-contained high-level language statements that performs a particular task and then returns to the caller. A service program contains one or more procedures. A program contains only one procedure.

The table below lists all the exported procedures found in the program object that can be externalized through this Web service. Expand the procedure row to change the default settings for the procedure parameters. The Usage parameter attribute affects what data is sent by clients and what is returned by the Web service.

Detect length fields **This time we allowed the wizard to detect the length**

Use parameter name as element name for data structures

Export procedures: ?

Select	Procedure name/Parameter name	Usage	Data type
<input checked="" type="checkbox"/>	RESTSRV5		
	REQUESTEDCAT	input	char
	RESULT	output	char
	PRODUCTLIST	output	struct

Select All | Deselect All | Expand All | Collapse All

Deploying an RPG Service (Contd.)

Note the returned structure omits the count field

- But it has obviously been used to control the length of the output
 - ✦ No blank array entries!

```
{
  "RESULT": "Found 1 rows for 01",
  "PRODUCTLIST": {
    "PRODUCTS": [
      {
        "PRODCD": "0000020",
        "PRODDS": "A duplicate test",
        "CATCOD": "01",
        "STOH": 456,
        "LNDCST": 20.00,
        "SELLPR": 30.00,
        "DTLCHG": 107,
        "DTLORD": 990105
      }
    ]
  }
}
```

Additional Options for Deployed RPG Code

Paris
Partner400

More sophisticated services may need additional data from the request

- For example when the query string is used to supply data
- Or the remote user information is needed

Select items in this list to make them available to your RPG code

Application Servers Installations

MSPTTEST - V2.6 (web services)

MSPTTEST > Manage Deployed Services > Deploy New Service

Deploy New Service

Specify Transport Information to Be Passed - Step 8 of 9

Specify transport information to be passed to the web service implementation code. ?

Specify Transport Metadata:

Transport Metadata	
<input type="checkbox"/>	QUERY_STRING
<input type="checkbox"/>	REMOTE_ADDR
<input type="checkbox"/>	REMOTE_USER
<input type="checkbox"/>	REQUEST_METHOD
<input type="checkbox"/>	REQUEST_URI
<input type="checkbox"/>	REQUEST_URL
<input type="checkbox"/>	SERVER_NAME
<input type="checkbox"/>	SERVER_PORT

Specify HTTP Headers:

HTTP Headers	
There are no entries for this table.	

Add Remove All

Back Next Cancel

Select any metadata required. It will be made available to your program

Last thoughts

Qshell commands are now available for IWS services

- They can be used to to define and deploy, stop and start servers and services etc.
- They can control some aspects in a more granular fashion
 - ♦ For example the URL used

Links to the manuals for IWS can be found at:

- www.ibm.com/support/pages/node/633935

IWS also includes the ability to generate code to consume web services

- But personally I don't find it anywhere near as flexible as tools such as Scott Klement's HTTPAPI

Questions ????

*Paris
Partner400*

**email me Jon.Paris @
Partner400.com**

**Join us at the final
Virtual RPG & Db2 Summit**

**[systemideveloper.com/pages/
events/RPGDb2Summit/](http://systemideveloper.com/pages/events/RPGDb2Summit/)**

